Universal Acceptance Day Uruguay - May 26, 2025

Technical Workshop – Configuring a Mail Server and a Domain Name Server with Support for Internationalized Domain Names (IDN) and EAI

Nicolás Antoniello – ICANN

Carlos Martínez – LACNIC

Nicolás Antoniello: Okay, those of you who aren't technically inclined, don't worry. What are we going to do now? What we're going to do now is going to get quite technical. What we're going to do now is, we're going to explain everything we're going to do, at least we're going to explain the part that we think you should remember about all this. Beyond the configuration details of the things we're going to do, beyond... there are even some things we're going to skip over, because explaining everything from scratch would involve, I don't know, spending a week here, let's say, just to get to what we're going to do today.

So, what are we going to do? What we're going to do is show them that everything we said this morning, everything that was discussed, that we talked about, everyone who spoke this morning, more or less... It's true. Right? We're going to apply all that theory.

So, what are we going to do? We're going to use this topology. This is a network. It's actually like many networks, but a network is also a set of networks. Right? And so, let's first explain what each of these devices is, what's installed on some of them, what we're going to install on some others, and what we're going to configure.

So, what we have here is this one that says CLI, it's like a specification. It's a kind of client. It's what a user's computer can be. Right? On that client, we're going to install an email client. Right? We're going to install PINE, right?

Carlos Martínez: PINE, Carlos. We're going to use PINE and MUT. Both. Two email clients. We're going to try PINE and MUT.

Nicolás Antoniello: There you go. We're going to install two. The old PINE doesn't work anymore and probably doesn't support any of this. The PINE sort of does. There it is.

So, what are we going to do? What are these two? They're email clients, like Microsoft Outlook or the email client we have on our cell phone, whatever it is, let's say. It could be a web-based email client, like when we use Gmail from the web. A program, an application, to generate and send emails. And, in turn, to access my inbox and read the emails sent to me.

We're going to install that on this device, which is a client. For simplicity, we're also going to need an MTA. Remember when we talked about the types of email servers? I told you there's one called an MTA, which receives emails and redirects them and passes them on to other email agents until it reaches the last one in the chain, which is the one that ends up depositing it in the recipient's inbox.

That would be like the successive mail carriers who receive the letter and pass it around until finally, I don't know, a small truck or a mail carrier on foot picks up the letter and delivers it to the recipient's home. That last mail carrier who delivers the letter to the recipient's home is also an MTA. And everyone in between is also an MTA. They're called MTAs in the electronic version of the mail, let's say.

As an MTA, we're going to install a very well-known, very well-known, and widely used one called postfix. Postfix, right? Or something like that, yes, postfix.

There's a first detail, let's say. Not all email clients support universal acceptance, and not all email servers that act as MTAs support universal acceptance. There are a lot of people here. I'm looking at Martín, because I know Martín is an expert in all this. He taught me many things years ago about what I'm talking about now, about the ironies of life. So, if you think I'm saying anything that's nonsense, correct it.

Participant (Martín): But we have always left people without mail.

Nicolás Antoniello: Ah, yes. We knew how to leave... We won't say, we won't say with Martín. We managed DINET's email in another time, in another life, let's say. Okay.

So, for simplicity's sake, we're also going to install that MTA, that email agent, on the same machine, right? In the real world, in a real-life scenario, or in a slightly more realistic scenario. A little more realistically, that MTA, in general, no, it's almost never or never on the same machine as the person generating the email, right? I don't have the agent that delivers the email on my device, computer, or cell phone, whatever, where I generate the email, right?

But here we're going to install everything on the same device to keep it simple. But that's still going to work. Why don't we install it on different devices? Because that would introduce other complications. There are some configuration complications that have nothing to do with universal acceptance, but that would take us a long time to configure to work properly, right?

There are network routing issues, there are reverse configuration issues for email, for the email agent to trust the sender, authentication issues, a lot of things we want to avoid. So, by installing everything on the same machine, we avoid all that stuff that's beyond what we want to cover today, right?

And there we have the email system installed, right? We'll generate an email with the client, and we'll send it to another user. We'll send it from one user, we'll send it to another email user, right? Using POSDIX, the MTA, the email distribution agent.

And using the email client or clients, we're going to use one client to send and another client to check whether we received that email correctly or not. And for that email, we're going to create an inbox, okay? With non-ASCII characters, okay?

We're going to create an inbox called Nicolás. Contain it, my inbox, okay? I want an inbox, contain it. I can't do that if the entire system doesn't support the universal situation, okay?

Okay, so we've got the email address sorted. But the email address, let's say, has two parts, right? The part to the left of the @ sign, which is the mailbox, is going to be Nicolás, and the part to the right of the @ sign, which is the domain name where the email server will be installed, right?

So, let's also use an internationalized domain name, right? In practice, in the demonstration we developed with Carlos, it's going to be Cigüeña, which has a lot of problems with ASCII-only domains, let's say, meaning it has the umlaut above the U and the Ñ. Okay? It's really internationalized, right? And let's see if it works.

But for the domain name part to work, we need to have a domain name system, an authoritative DNS server that supports that and has the domain created, right? And we need to have a recursive DNS server, which are the search engines within the DNS, that also supports the universal situation and recognizes the universalized domain when the email client and the MTA, etc., to everyone who needs to make the DNS query, make the query, tell me what Stork's IP address is, right?

If they don't understand non-ASCII characters, they won't be able to figure that out. So, before configuring the entire email component, we need to configure the entire DNS component.

For the DNS part, we set up a super-real scenario, right? In fact, maybe we should have made it less realistic because it would have taken less time, but it doesn't matter. This one is really, really realistic. This is for the more... technically inclined. For the rest, don't worry too much about this.

We have an authoritative DNS server, which is going to be the DNS server, it's going to be the authority for the... It's going to be the one that has the IP address information corresponding to that Stork, right? Which is what's called a hidden server, let's say. This server isn't going to be publicly accessible, right? And we named it SOA, right?

And then we have two public DNS servers, which are the ones that will see what they consume, when I want to resolve the domain Cigüeña, right? This server uses authoritative server software called BIND, BIND, BIND. BIND 9, because it's the version from years ago. There was always something that wasn't BIND 9, well, there

was, but I never used it. Whenever I used it, it was always BIND 9, let's say. BIND 8, I never used BIND 8, that's why.

This SOA runs a BIND 9. This... This... This... This... This... This one is also a BIND 9 and this other one is an NSD. There it goes, this other one is an NSD.

And then we have to have this recursive server here that runs recursive server software called UNBOUND. Then we have a bunch of DNS software that we'll have to configure to support this universal acceptance, and also add the Stork domain configuration, etc., etc., and make sure all of that works.

So, let's first configure all the DNS settings. Then we'll configure the email client and the email MTA. And then we'll run a test, or as many tests as you want. And we'll see if it works or if all of this doesn't. And we'll do all of this in less than two hours.

To give you an idea, and we were surprised by Carlos, I think we did it in 15 or 10 minutes. A little before lunch. But let's see if we can do it again so you can see that it's not impossible. You do need technical knowledge. I mean, we did it quickly because we do this often. But with the right guidance and the right knowledge, it can be done very easily.

And you'll probably have to do much less than this. Because you'll see that, for example, email servers don't require any special configuration because they already natively support universal acceptance. So, all we'll have to do is a basic server configuration and simply add these universalized domains. And we'll see how they're added, in what format, and so on.

All that talk about PUNY code, let's see what works in practice and how to configure it. So, what I'm going to do is, I'm going to do the DNS configuration first and show it to you. And we're going to chat there, among you, Carlos, and me. We're going to ask all the questions you want, and any questions I don't understand that you want. And then Carlos is going to do the same for the email part. And then among everyone. We're going to run some tests, send the email, and cross our fingers to see if we receive it or if this is all a hoax.

And no, it's not a farce, but anything can go wrong, so to speak. If this has failed so many times for Mr. Bill Gates and Elon Musk, why can't it fail us? It's a good excuse.

Okay, so what I'm going to do now is, first, the super-technical questions are there. This isn't software. It's commercial software, so to speak. We designed this entire

lab environment; I designed it with some colleagues at ICANN, and it's the one we use for all our labs.

To the horror of all programmers, it's all done in Jelscript, let's say. Horrible, frightening, horrible. With all the bad programming practices possible, because I'm not an expert programmer, but it works. It works and it scales. We've tested it, it's passed tests that we never thought it would pass, but well, yes, yes, yes, we can run, I mean, each one, all of this runs on a virtual server, each one of those is a virtual machine, it's an LXE container with Ubuntu Server fully functional and we've tested it for more than 200, more than 350 containers running simultaneously and doing routing practices with Carlos, full routing, etc.

I mean, it all depends on the size. The virtual machine size, let's say that, but oh well. Okay.

Participant: What was the first art that was one of the events?

Nicolás Antoniello: The one from the ACNI events in Panama. The ACNI in Panama had more than 400 containers and more than 400 routers, that is, this entire infrastructure replicated 400 times, each router with partial but real routing, let's say, the entire internet routing table injected there, and it didn't explode. Good.

So, the first thing I'm going to do is configure the DNS, okay? Obviously, I have the pre-made settings here. I'm just going to paste them in and explain more or less what I'm doing, okay?

So, here, on this DNS server, well, this is better with the headband, but anyway. I'm going to change to the BIND directory. Okay. Okay. I mean, for those of you who aren't that technical, don't worry about understanding everything that's going on. Let me put the microphone down.

Can you hear me there if I speak? Like that? Can you hear me? No, I can't hear anything. Well, wait a minute...

Okay, for those who are more familiar with DNS, the first thing I'm going to do is create a zone, a file. A zone for the authoritative DNS server, which is in that file that is created. That's part of the DNS server configuration. I load all the DNS information I want to store on that server. Specifically, it's in this file that we'll later put all our domains. Specifically, we're going to put that universalized stork domain

on this server. But first, we need to create the file with all the basic information necessary for that server to work.

So, this is going to be the name of... The domain name of my server, grp1.uamvd, we named universalappsetamontevideo, and this ending, tlabs.training, is the ending of the domain name that we're going to use for our practice.

So, now I'm going to paste a configuration, and I'm going to quickly walk you through some parts of the configuration that are relevant to what we're going to do today. Basically, this is a basic configuration. A zone file. Like the domain, remember, it's grp1.uamvd.tlabs.training. So, I need to change it here, right on the side where it says X, to a 1.

This, again, for those who aren't technically savvy and don't have experience with DNS, don't worry. I know it's difficult, but... And for those who are familiar with DNS, well... I'll explain what I'm doing here a little bit in the same way now.

So, what I did here was configure a DNS zone file, for the more technically inclined. What I did was configure a zone file. These are the parameters of the zone's SOA record, which are entirely ours for what we're going to do today. These times are super low, compared to almost any time used in reality. But we want things to run very quickly in the lab when we make a change. So, we don't want to wait. Horribly, like weeks, until this works. So, we want the changes to be very fast. That's why we set times in very low seconds, let's say. Something like 100 or 1,000 times less than what one would generally use in production.

These are the two public servers, the two public authoritative servers, NS1 and NS2, that we're going to use. They were in the diagram, if you remember, in the diagram. The server we're configuring now... It's this server, the SOA, which is going to be hidden. And NS1 and NS2 are the public authoritative servers, which are going to be public. Exposed for anyone to query, because an authoritative server that can't be queried is useless. By definition, all authoritative servers have to be public if I want them to be queried from the Internet. Okay, then...

Participant: Can I use the headband?

Nicolás Antoniello: Wow! Every time I did this... Hello, hello, hello, can you hear it there? Wow! I sound like Jorge Real, all things considered. It's not that I want to be like Jorge Real, but, well, I always see him wearing this headband, and it's cool.

Okay, so let's configure the SOA now, okay? As I was saying, this is the zone file configuration, and then these here are the IPv4 and IPv6 addresses. The lab runs dual stack, IPv4 and IPv6. This is the address. These are the servers, NS1 and NS2, the public authoritative ones, the ones I was telling you about, okay? And this is the IPv4 and IPv6 address for each of them, okay? A super basic configuration of a zone file to get it more or less working, okay? Okay.

So once we have this here, let me check to make sure I didn't... That I didn't forget to change any group numbers. Oh, here, yes, nothing's going to happen, but we're going to do things right, it won't cost us anything. There, there's an X there. Also, to the horror of everyone who programs and configures things, I'm using Nano. Yes, I'm still one of those who uses Nano. And that's it.

They should be there, so let's do a... You can come back anytime to do this, Martín, thank you very much, because things get tough after lunch. There it is. I'll restart the BIND process so it takes the configuration, the configuration changes I just made.

By the way, that initial BIND configuration we had, to which I added what you saw there, the zone file, that is, this initial configuration of... This doesn't happen to Real. It doesn't happen to Real. No. This configuration, BIND is configured, it doesn't have any configuration other than the one that comes by default when you install it. I mean, all we did was install it, and we're doing the entire configuration from scratch now. It's not like there are any pre-configured things; everything there needs to be configured after you install it, so we're going to do that now. Okay?

So, we create the zone file. First thing, or you can do it in any other order, but anyway, I like to create the zone file first. I actually rushed to restart the server because I need other things to get this working. So, now I'm going to edit another configuration file. Name.conf.options. I also need to add configuration to it for this to work. How we want it to work.

I'm going to get this out of the way. Again, this is for the more technically inclined. This NSSEC line, Validation Auto, doesn't even need to be included because this isn't an NSSEC practice. But anyway, it's NSSEC these days; we should all have NSSEC implemented on our DNS servers. So, I'll leave it there. We're not going to sign the zone anyway; we're not going to do anything related to NSSEC. So, if I don't include this, it doesn't matter. And also, if you don't include it by default, it's automatic, so bye. Let's forget about this.

What matters here is that, since this is an authoritative server, I don't want it to run as a recursive server. I tell it not to run recursion. That capability, which is the most important. And this one that says it should also work with IPv6. Because if I don't set that, it'll work by default. Well, I have a question about whether all versions work with both by default. Or only with IPv4.

Participant: No, by default, if you don't add that line, it's only IPv4. But that line is already written in the configuration.

Nicolás Antoniello: So, if you don't change anything in the configuration file, nothing happens. In short, the only thing you should add to the configuration, in this file, in name.com.options, is this last line. Go to the end and add recursion, no. Regarding what's already configured by default. That's the only thing you have to do to tell it, this is an authoritative server and I don't want it to also be a recursive server. Because Vine can be used for both things. Right? And since it's not good practice to mix authoritative with recursive, nothing. I make sure it doesn't work as a recursive server. Right? Good.

And there's one more thing left to configure. That's name.conf.local. Another configuration file I need to edit is name.conf.local. Which is empty. All of these are comments. Right? So, what I'm going to add is some basic configuration as well. Basically, what we're doing here is telling Vine, we already created the zone file. What we're telling Vine here is, what is the zone file? And how do I want the DNS server to behave using that zone file?

So, here I declared the zone. My zone is going to be called grp1.umvd.tlabs.training. This is the name of the zone file we created a little while ago. I commented out these lines, but I'll leave them here in case anyone wants to replicate this later. Those inline signing and auto-dnsc-maintain, if you configure DNS and have DNS enabled, which would be desirable, which we should all do. Uncomment those lines, and the server will automatically comment out each time you make changes to inline signing and auto-dnsc-maintain, once you generate the key pair.

This is Chinese for those who don't know DNS. Once you generate the public and private keys with those two lines, every time you modify the zone file and reload, restart the server process using rndc-reload or system-control-restart, automatically, as those two lines are set up, it re-signs itself, without you having to do anything. It re-signs all the records you've added to the zone file. Okay? Okay?

You have DNS all by yourself, without you having to do anything, magic, right? And for those who don't want to use that, it's just a matter of time. You're going to have a problem because you're going to forget to sign something and things will get messed up... That's one of the reasons why most people think DNS is complicated, because they want to do things manually. This was invented years ago, it works great, and I only have to worry about checking it every so often to make sure everything is okay, but the system takes care of everything on its own.

That is, I've commented it out now because we're not going to do the DNS part. What matters is this: it says typemaster, which means this is the primary server, which will be hidden and will pass the information to the secondary servers. This is the same thing: this tells it where the private key used to generate the DNS signature is. Since we're not using it, I commented it out here.

This is the name of the zone file, so to speak. It was the name I assigned to the file when we created it a while ago. I made it the same name as the domain and added .zone at the end so it's easy to identify. And then this, allowTransferAny, is something that isn't recommended in production. Instead of the "any" inside the allowTransfer, you have to enter the IP addresses of the secondary servers, which are the only ones I'm going to allow to transfer the zone so that just anyone from anywhere in the world doesn't come and steal the entire zone.

Basically, by setting up XFR, they could steal the entire area as is, but since the lab is all closed off in a private environment, no one can access it from outside, no problem, and we make it easier, we simplify it that way. And that's it, right? Now, there it is, now yes. There it goes. And now we restart the... we restart BIND, the server. Let's verify that everything is okay. No, everything is fine.

So, what we did now was modify three configuration files. In one, we created the zone file; in the other, we configured BIND to be an authoritative-only server; and in the other, we declared the characteristics of the zone file name it creates and how it will behave as a master, a primary server, so to speak.

By the way, that thing about master, before, in another era, many years ago, the language used was this, master and slave, let's say. But master and slave have negative connotations from a social and cultural perspective. So, for many years I've been almost certain that if I put primary and secondary, everything works perfectly, right? I left master there because I left it as it was by default, but in general we didn't use master and slave anymore, we used primary and secondary, okay? Fine.

Participant: How do you set up the reverse?

Nicolás Antoniello: There it is. No, I'm not going to configure it in reverse. If it were when we were going to configure the MTA, the email server, the email agent, in general, a very common check that MTAs perform is that the reverse exists and matches the direct one. Again, Chinese for non-technical people. You should definitely do this when you have something in production and you have an MTA on a different machine than the client, etc.

Since everything is on the same machine now and there won't be any outward routing, if it's outside the email machine, it won't check that, and there won't be a need to configure reverses. You can also ensure that Postfix doesn't check that and just blindly trusts it, and nothing happens—it's the same thing.

But in a real-world scenario, it would be advisable to create the reverse zone, serve it from the same authoritative server, and configure the email server's reverse zones so it works properly. Another thing we're not going to do is create an MX record. We could do that, but again, since we're not going to send from another machine, we won't need the MX record. But if you create an MX record and you're in a production environment, it's almost certain that you'll have to create the reverse zone and configure it correctly. Otherwise, at some point you'll run into problems with a mail server. But since all of that doesn't exactly have to do with universal acceptance, we'll leave it out. But yes, the assessment is correct.

Okay, the authoritative server part is now configured. So now we need to configure the machines with these two—this will be quick—on NS1 and NS2, so they copy and transfer the zone we just created and become the servers that will serve or make public each of the zones.

I'm going to the BIND directory, because this secondary server is also using BIND. And there I'm going to configure—there's just one file I'm going to have to configure there, which is named.conf.local—so that this is a secondary server.

We edit named.conf.local, which, as you can see, doesn't have anything. It's how it appears after installation. As it comes from the factory, so to speak. And I'm going to add this very simple configuration. I'm going to tell it that the zone I'm going to serve is going to be the same one we generated on the other server. It has to have exactly the same name: GRP1, UAMVD, TLabsTraining. Type, here Slave, here you

could put Secondary. I'm not going to test it now because if it fails we'll waste time. But remember that you can set Primary and Secondary.

And I have to tell you what the IP address is. It can be version 4 or version 6. Here, to make it easier, I just put the IP address version 4. But you can put both. It's where the primary server is from where the zone will be copied. That is, the IP address of the SOA, the SOA server. In this case, it's 100.100.1.66.

And that's it. This should already be working as a secondary server. Of course, don't do this at home. If you set up a secondary server, you have to configure encryption and authentication so that not just anyone can become your secondary server and cause havoc by hijacking your zone or anything else. But security isn't what we're going to focus on here today either. So, that's it.

We check the configuration and reboot the system. Restart. Good. Good, there it is. It's that easy. It's that easy, or not. Or that difficult. The secondary NS1 is now configured. Let's configure NS2 now. This isn't a BIND. This is an NSD. No, this is a BIND. No, it's NS2. The other one. The other one. There are two. NS1 and NS2. The secondaries. These are secondaries. Of course. This is NSD.

There it is. Slash etc. Slash NSD. And here's another little file too. Guess which file I'm going to modify: NSD.conf. That little file only has this. We're also going to add the same thing we did to the other one in the NSD format. Which is a little bit more. You have to add a few more things for this to work. But anyway, it's basically telling it where it's going to save the zone file. Telling it the data so it can transfer the zone from the primary. What the IP addresses of the primary are. And the name of the zone it's going to serve. Which has to match the primary's. Yeah? And that's it.

There you have it. Then this. We'll leave all this for you to have. Things that happen in production. I'm going to edit it again. NSD.conf. And now I'm going to paste it again. And I'm going to modify it. I was doing it with a user who doesn't have the privileges to do this. Well, of course, another thing you shouldn't do is configure all these things with the administrator or root user like I'm doing. Instead, the appropriate thing would be to create a specific and exclusive user for the DNS server. And use that user for all DNS software configuration. But again, we're not doing a security demonstration. And that would take more time.

Okay. Ah. There it is. Done. So now we have our authoritative server configured, which is the primary SOA, and the two secondary authoritative servers, which are

the public ones: NS1 and NS2. Let's quickly configure the recursive server. Yes. What is Unbound? Like I told you. Unbound.

Conf is the file we need to modify. And here I'm going to add a basic configuration for an Unbound recursive server. I won't explain this. Those who are more technically inclined can look it up later and ask any questions they want online. But basically, yes. I'll explain it a bit.

These are the server interfaces that I'm going to use to listen for DNS queries and send DNS queries. When I put this like this: 0 0 0 0 0. In IPv4. Or 2.2.0 in IPv6. I'm telling the server to use all interfaces. Since it only has one interface. Done. That's it. If you were in a production environment and you have a DNS server that has a management interface and a public interface through which you want to route DNS traffic. Here you should only put the IP address of the public interface. Not the management interface. Because otherwise, I'm also going to listen for DNS queries and perform DNS resolution through the management interface. That's it.

This here is access control. I think it's the newer versions of BIND and Unbound. If you don't put it in Unbound at least. If you don't put these things, the server won't respond to anyone. Basically, it won't respond at all. So I necessarily have to consciously tell it who I'm going to serve with this server. And I put it here. This is the this. What it calls localhost. So it's so the server can respond to queries made from the server itself. And these two here are the address range.

And you can leave it out. I always do. But by default it's going to be port 53. And if you change it, it won't do any good. Because no one will know what their server's port is going to be. So, in general. I know of two cases where someone changed this. But it's a very specific case. And it's not a public DNS server. And this is so the recursive server does TCP, UDP, and works with IPv4 and IPv6. That's it. Ready. Nothing needs to be changed here. Let's see if it works. We're there. Ready. It's working now.

So we already have an ecosystem. We have a recursive server. A hidden authoritative server. Two public authoritative servers. And we haven't even started with universal acceptance yet. This is just to get you started. So now what we're going to do is the following. We're going to go to the client. We're going to go to this client, which is where we're going to install the email servers. Yes.

So I'm going to access the client. And what I'm going to do in this client is tell the client which DNS server I want it to use. So that we can do practical work on the real

internet, let's say. The server that's configured by default is a server installed on the lab platform. And it sort of relays to a public Quad Nine server or something like that.

We have to modify the resolve.conf file, which is in the etc slash etc slash resolve.conf directory. Yes. And instead of putting 164.01, which is the address of that lab platform server, I have to put the IP address of the recursive server I just configured. Yes. From Unbound, which I don't remember which one it is. So, and conf. And we're going to put only the IPv4 to make it faster. It's 100 at 168. Yes.

So I'll use the recursive server I just configured, which is the one at that IP address within the lab. Done. Done. That's it. And now let's test that this works. Let's run a query with dig. It was NS1 dot GRP1 dot UAMVD dot TLabs training.

I'm querying the recursive server for a domain created on the authoritative hidden server. That domain is going to be transferred. The file is going to be transferred to the secondaries. The recursive server is going to query some of the secondaries and should get a reasonable response. That's well written. Yes. Good. And there it is. Yes. I made the query in NS1 and it returned the IP address associated with what I queried in NS1 point GRP1, which is 100 at 1.130. Okay. Apparently this works. Yes.

I ask for the other server, NS2, which was another domain that was configured. It'll return 131. It is. And what are we going to do now? Now we're going to do the last thing we're going to do on the DNS side, which is add our domain stork. Yes. So I'm going to the SOA to the server, the first one we configured. The authoritative one where we have the zone file. We edit the zone file. Yes. And we add the universalized domains we want to use. Which are, we're going to create two. We're going to create one called stork and another called cannon, which also goes with ñ. So it's also universalized. Yes.

So we add those two things to the zone file. We add them. You can't see anything back there. No. I just realized you need a telescope to see that. Of course. But why didn't they warn us? That's proof that we're all asleep after lunch. Let's see how this is done. There it goes. There it is. But now what I have to do is because now the whole environment is gone. Wait. I'm going to open the SOA again so that these things that happen with the screens when you resize things. There it is. Good.

So what do we add now? We add this. We add, we add. It's a comment. We add this domain, stork. Yes. What I put here is this, here, XN dash dash stork dash. RTA9E is

the Puny Code for stork. All Puny Code anywhere that starts with XN dash dash already knows that it's an encoding of a non-ASCII domain name. Yes. Encoded with the corresponding Puny Code using UTF-8, etc., etc., etc. Yes. This is how you say stork in Puny Code. Yes.

This is the A record for the version 4 IP address corresponding to Stork. Stork is going to be the name of our MTA, our email agent. Yes. Since we're going to install the email agent on the client, I have to enter the client's IP address there. I'm not going to search for the client's IP address either. Here and with ifconfig.

Do you remember what I was telling you this morning about the conversion rules for converting non-ASCII characters into UTF-8 byte sequences? This is a conversion rule that also applies to the result. Puny Code is also a conversion rule that also applies to the result.

The issue is that not even all ASCII characters are valid in DNS names. There are characters that can't be used. The period is obvious. The period has a special meaning. So you can't use it as part of the label. The same thing happens with the underscore, for example. It's reserved for certain specific uses. So, for that, the UTF-8 conversion rules weren't sufficient. So they created these Puny Code rules that transform UTF-8 into something digestible by the DNS.

Which are very basic. They're lowercase characters. Not even capital letters; it doesn't really matter if they're uppercase or lowercase, they're equivalent. Numbers and some punctuation marks, but not many. In fact, I can't use the hyphen as a separator either. And one curious thing about Puny Code is that what it does is it says stork there, which reads stork.

Well, what did he do? He removed the special characters, and what comes after the second hyphen is the encoding. What comes after is generated with an algorithm that encodes the Unicode code of that character plus the position where it should be inserted. The U with an umlaut and the position of the umlaut are encoded, plus the ñ with the position of the ñ. Everything from this one here to this one here is the encoding of the ñ. What I'm saying is, let's say this one contains that it's one, and it contains two encodings, and this one contains only the encoding.

I have to look at how you might wonder why the accent ñ is right in such and such a case. There are some optimizations to that, and the curious thing is how those characters that come after the hyphen are derived. It's a base-36 numbering

system. No, it's not even hexadecimal. It's base-36. All of this is good, as always, as we do every last minute with me, this one as it should be.

When we did this last year, someone asked me what the Puny Code rules were, and I honestly had no idea. I'd never even bothered to read them. I knew what they were supposed to comply with, but oh well. In DNS, short is twice as good for multiple reasons. Because later, if I have to sign, it's easier to sign responses that come in a packet of less than 512 bytes. That includes the header and all headers that are less than 512 bytes.

The reason for this is to be able to transmit it in a UDP packet without having to use TCP. While DNS can use TCP, and in fact, if UDP fails, everything is optimized to the maximum, it's about making the maximum effort to ensure responses are short, concise, etc. So you see that in computer science, there's a trade-off that always exists. If you want to save memory to save memory and storage, you use CPU, and if you want to save CPU, you do so at the expense of memory and storage. And you can see a bit of that in action.

Participant: Oscar, is there a question in the presentation about our school this morning or in a previous one? No, no, let's see, oh, no, she knows, I know where it said it, I don't remember.

Nicolás: Of course, exactly, because regular expressions are a mess in ASCII, and regular expressions are an atrocious mess. Regular expressions in Unicode are much worse. So what that slide recommends is basically trying not to go all out and create your regular expression to validate emails. Instead, use an existing library. You can even install packages for operating systems that give you command-line tools to generate Unicode for whatever you want. Yes.

If we use a page for what's called Punicode. Punicode. There we generate the stork and the cannon, let's say. But yes, in a comment to add what Carlos said about the length of the encodings. In DNS, there are always a lot of reasons why, as Carlos said, we always try to minimize the amount of information transferred. That has many advantages from many points of view. And today, perhaps the least important is bandwidth, let's say, one of the least important.

There are security reasons for doing this as well. To reduce the capacity to be used as a weapon, so to speak. If it responds too much, it can be used as a weapon to carry out an amplification attack. This is why, when a public key is generated and something is signed in DNSSEC, what is actually signed is the hash, not the

information itself. Because otherwise, the encryption of a text, so to speak, would be monstrously enormous, etc., etc.

Surely, if you're here because you've had contact with the Internet, you all know there are thirteen root servers. Someone, someone, everyone knows there are thirteen root servers. Have you ever wondered why thirteen is a rather unusual number? Everything is always a power of two. Not how much less than even, at least not even. Well, thirteen names.

Carlos: Exactly.

Nicolás: Well, the reason there are thirteen is because they fit into a 512-byte packet. So it's something that comes from super immemorial times and makes a tremendous effort to ensure that the greatest number of DNS responses fit into a 512-byte packet. Yes, in fact, as Martín said, today throughout the world, their hierarchical system is a hierarchical database. The information is distributed, and that information storage distribution hierarchy has a higher part of the hierarchy called thirteen servers than originally thought. In reality, there are more than nineteen hundred copies in the world. But to maintain the thirteen, an anycast technique is used in which those thirteen are duplicated, multiple copies of each of those thirteen are generated. So, they are still thirteen different IP addresses, but multiple copies of each of them, all with the same IP address.

Don't follow the rabbit, as I was saying, they're going to ask, but didn't you mention a little while ago that IP addresses have to be unique? Okay. So, Stork is encoded like this. This is the IP version 4 address and the IP version 6 address of the server where our server, our email agent, the MTA, is installed. That's the software we said we're going to use, Postfix.

100.100.1.2 is the IP version 4 address of the client machine where we're going to install the server. And the other FD three A of four zero nine two dot one two dot dot two is the IP address 6 of that same server. And we created this domain just for fun, and so that it wouldn't be just one, we created another one that encodes like this, and we created an A record with a private address that isn't assigned to anything, so we can run the DNS query through the domain and see if it actually responds with that address or if it works. We haven't set everything up yet.

So that's what we added to our zone file. And here, as always, I forgot to increment the serial number. And that's why we technicians racked our brains for hours trying to figure out what was wrong, and it wasn't wrong at all. You just forgot to

increment the serial number. And if you don't increment the serial number, no matter how many times you restart the server, it won't make any changes. We restart the server, reload the configuration, and that's it. Now we go to the client.

Remember that from the client we had changed it, we told it to use the configured recursive server. Now we add things to our server, what we had said, we did this DNS query, and let's say it was working fine, responding. But now we add two domains, let's see if it works. In query by NS1 GRP1, we're going to query by stork. Oh, how do you spell stork on the English keyboard? Well, if GU doesn't, don't do this at home. At Herrero's house, my machine is, oh, I'm realizing that ICANN's machine doesn't have ñ, with me, they're not universal.

Okay, no stork, let's see what's going on. There it is, and there's the IP address for stork. If we ask for the AAAA record, which is the IP address version 6, it should also resolve it there. And if instead of asking for stork, we ask for cannon. I know how to do this. There it is, too. There it goes. It answered with "no." It answered with nothing because I didn't put it because I didn't put AAAA record. Let's ask for the A record, and there's the IP address that we had configured.

So our DNS with universalized domains, the entire DNS ecosystem, is working perfectly: the recursive and the three authoritative ones, the hidden and the two secondary ones.

A question I've sometimes been asked is whether when I use Puny Code, when I use the real character, there must be some technical name for what I call the real letter. The reality is that for DNS, the only thing that exists is Puny Code. For those who translate those characters, the real labels, into what would be Puny Code, for example, DIG, notice that in DIG, Nicolás is passing it as a parameter the name written as we want to see it. DIG is internally converting it into Puny Code, but if I use Puny Code...

Carlos: The DIG doesn't care.

Nicholas: This is what you shouldn't do. You see, when you grab something, there are two things you shouldn't do. One is to use the organization where you work as an example. The other is what we're doing now: one says something and the other comes down and wants to verify it, but let's see...

Carlos: He's just getting into it himself. I didn't tell him to try it.

Nicholas: I'm going to try it, I'm going to try it with cannon, instead of putting cannon, I'm going to put this thing here that only Neo understands. You can put the internationalized characters or you can put the Puny Code, which might be useful in some circumstances if you have limited access to, I don't know, a keyboard that has those characters.

Carlos: You are right Carlos, I confirm that you are right, it works, that is, if I put cannon or I put the Puny Code it works the same because the DIG, the DIG application that makes the DNS query, as Carlos said, what it does is if it detects the Puny Code it sends it as is, if it detects something that is not ASCII, the first thing it changes is for the Puny Code and what the DNS query sends is the Puny Code.

Nicolás: A bit like what happens with reverse queries in the DIG, you put in the IP and it just transforms it into the query for the zone of that INADR ARPA, but if you ask it for the name of INADR ARPA, it also does the normal query.

Carlos: Chinese not to have...

Nicholas: But someone asked about the reverse, that's why I say...

Carlos: He realized because...

Nicholas: Exactly, and he realized it because it starts with XN...

Carlos: Each Puny Code name has that, it's like a little packet in which it has a start mark and some delimiters.

Nicholas: So, in addition to all of this, of course, since the invention of this Puny Code and universal acceptance of coding, there can't be any domain name that begins with XN, because otherwise it'll be thought of as a Puny Code and it won't work. So that's a prohibited coding for a domain name.

So the changes that are made to the DNS, well, you at ICANN suffer much more than we do. There are so many, so many, so many DNS servers and so many domain names created that any change or any decision that is made sometimes requires years of data to be taken in order to decide...

I probably don't know because I wasn't part of it, but how they came to believe that XN was a good start delimiter... They probably collected data for a long time to be able to see that if it was used, it was a limited use.

There's a lot of data collection that's been done on this, for example, to see what names people use on private networks. People who have private networks, large banks, companies with many branches, use their own internal DNS and give them random names, not caring whether they're zones that exist or not. And often those names escape onto the Internet. Because you have one or two poorly configured machines that, instead of querying the company's DNS, query a DNS on the Internet.

So that's what they call leaks, name leaks. The problem with name leaks is that these leaks eventually confuse others and can also be used for attacks. There's a famous case where they used the company name .corp. They had invented .corp because they knew it looked nice and they liked it. But what happens? It's a pretty obvious idea. So they didn't come up with it alone. A few hundred other people had it. So they started confusing each other. So all these so-called collisions generate a lot of problems.

So there's a lot, and I'm often impressed by the detail ICANN goes into. They collect and measure data for years in some cases before making a decision. No, well, we're going to use this string. Because it's the least bad. Generally, it's not because no one used it, but because those who did were very few, as you said.

There are hundreds of strings that are blocked. For example, localhost. There can't be a localhost TLD because there's a lot of filtering of information that goes out into the world, like a DNS query, let's say, through localhost. Of course, a new TLD with localhost isn't possible because that one is blocked.

And in fact, there are a ton of security measures that have been added, like Autonomous System 102, etc. Because allowing that is also a very bad practice. Because it can be used for a ton of code injection attacks. Because you can send a DNS query that crosses almost all firewalls. And when the machine reaches, as you say, localhost, the machine will interpret it as something very different from what you think. And it can execute local code. You can do anything.

Carlos: Going against my advice not to follow the rabbit, what Nico from AS112 just said is a tool created on the Internet to try to function as a sort of vacuum cleaner for all those strange DNS queries. Because they generally have the characteristic of asking, for example, for the reverse side of private IP addresses. That's a classic. People configure that on a private network and have DNS leaks. But those packets originate from the company's private number. So, on the Internet, they reach their destination but then do nothing. Not even the servers try to resolve them.

Nicolás: Of course, they try to solve it, and they take up a lot of load. So, AS112 was invented to try to extract those queries. For example, AS112 in particular is for reverse queries from private IP addresses, for example, 10.

Carlos: All these things seem, especially when we look at it from Uruguay, where everything is small here, everything is very quiet, nothing much ever happens. Well, the AS112 nodes—who's going to miss a reverse query on the 10? Well, they receive gigabytes of traffic.

Nicolás: And they're also blocked with this universal acceptance. For example, one of the first written languages—I think it was the first one implemented in the entire Puny Code dictionary—was Chinese. It's also the most widely spoken language in the world. I think the first is Chinese, and then Spanish, or something like that. And of course, what about localhost? Written in Chinese, it also has to be blocked because it's normal for an operating system that's in Chinese, by accident or mistake, to use that encoding.

So all of this also generates a ton of domain names that also need to be taken into account for security and load issues. It's like everything else. Technology is fantastic, it's wonderful, and it enables all of this we're talking about now. But it also opens new doors for those who want to engage in malicious activity. It's like everything else. So we have to generate a ton of countermeasures and a ton of protection mechanisms that weren't necessary before simply because they couldn't be done. It's like two sides of the coin, technology in general, and everything we do.

Okay, now Carlos will explain it to you in half an hour.

Carlos: Is it my turn in half an hour?

Nicholas: Well, let's try. We'll try.

Carlos: Well, I'm going to have to stop sharing. In half an hour or in five minutes?

Nicholas: In half an hour?

Carlos: Yes.

Nicholas: Perfect. I was really scared for a second, so to speak.

Carlos: Yeah, yeah. I'm going to have to stop sharing. Maybe I've even left without speaking. It's all about promoting the myth, isn't it?

Let's create the users. Here comes the first thing that's strange. No, it's not strange. Let's create a useradd. This is Ubuntu 20.04, I think. 20.04, yes. 20.04. There it is. But basically, these things haven't changed in the last 30 years. I create them with the bash shell because it's convenient for me. But that's up to you. Let me create the home page, and this one will be called Nicolás. There goes Nicolás. And I'm going to create another one called Martínez.

For those who are used to configuring the email server, we're creating operating system users here. When you create an operating system user, if you have Postfix installed, for example, the mailbox associated with that user is also automatically created. Right? It's an easy way to create a mailbox. We could do it by creating the mailbox directly in Postfix, but again, the practical effect would be the same; it wouldn't contribute anything in terms of universal acceptance, and it would take us perhaps half an hour or 40 minutes, plus practice. So we do it directly in the operating system in useradd, which already supports universalized characters. I can now create directories with non-ASCII characters, etc., which, as you may recall, wasn't possible before. Before, the document name couldn't have an ñ, couldn't have an accent, couldn't have... I'm from the era of eight characters plus the three extensions, right?

Nicholas: Yes, worse too. My baldness shows it.

Carlos: Sorry to those on Zoom, I didn't notice. I started talking loudly, thinking we were all here, and I forgot about Zoom. Oh, well, the users were created there. This is going to be a difficult topic.

Nicholas: Change of headband.

Carlos: Change the headband here.

Nicholas: Yes, it's a change of headband, isn't it?

Carlos: Yes, yes. Change the headband. It doesn't go like that, it goes backwards. See? This is a problem. I'm going to stick it in my windpipe. Backwards, let's say. I'm going to stick it in my windpipe. There it goes, there it goes. I've done it before, I put it in my little pocket here. There it goes. It has some stretch to it too. Come on. Oh, look. I'm more stubborn than Nicolás, that's what has just been demonstrated.

Nicholas: I didn't mean to say it.

Carlos: Well, the users were created here. Remember, there's no Puny Code here. Puny Code is for DNS. What we have here is something that happened in the late 90s, which was the conversion of operating systems to use UTF-8 as the character set. One interesting thing that Nicolás said is that we're using—if you will, we're creating a mail system here—what would be a traditional UNIX mail system, in which mailboxes are linked to the operating system users.

In modern email systems, like Zimbra Outlook, this isn't the case. The concept of mailboxes and email system users is no longer necessarily related to the operating system users. I don't know how many it has now; I think it has many more than when we were here. There are not 300,000 ADINET users in a UNIX system. Those users exist as a mere figment of a database. But, well, this is a bit old-school. And there are many places where this is still done because it's very practical. It doesn't require paying anyone, and it's done with a few commands.

So, we have the users. Great. Now we need to install some software packages. So we'll do that with a few commands: apt-get update first. This will download the list of packages. And now we'll click...

Nicholas: Oh, Carlos, you never update this. It has 139 updates.

Carlos: We'll download it like this. At your own risk, do an update. Do an upgrade. When there are more than 100, you have to leave it like this. The machine is already becoming vintage.

Nicolás: I'm going to do a... While Carlos installs the software, I'm going to explain something that has nothing to do with what we're doing today, but it has to do with the platform. Actually, this platform, the operating system that's happening now in containers and outside of containers, is Ubuntu 20.04, which has already ended support, or will end support now in June. At the beginning of June. And we already updated it to 24.04. I think it's the next one that has long-term support. But obviously, never try to do a lab with something you just installed. So we're still using an old lab. So I don't recommend upgrading because it's basically going to change everything.

For many years, the May LACNIC event coincided with the release of that year's macOS. So... What happened? There were several LACNIC events during which people took advantage of the event's internet connection and decided to update their Macs. And they ended up, of course, when it wasn't working, as was obvious,

and they came to us to ask. Often, we have nothing to say to them. I mean, format it, I don't know what.

Carlos: Okay. Here's what you're asking me about installing Postfix: Do you have two or three...? Let's see, you could install Postfix... It's an open-source program. You could install it from the source and configure it from scratch. But all these Ubuntu packages already come with some configuration templates that make your life a little easier. The Satellite System? You know, I've never been very clear about the Satellite System. I'm pretty clear about No Configuration. You have to do it yourself. And what we need is the Internet Site. If you were going to configure an email system for yourselves... I have one that I use for this kind of joke. What works on today's internet is this one. It's the Internet Site with Smart Hosts. We can talk about that another time. I don't want to go following the rabbit hole. Internet Site.

Here we have to tell you... All email programs have, like many other identity problems. They need to recognize themselves. Find themselves. So when they have a single name, it's relatively easy because that's what you're asking me here. What's my name? Who am I? You're asking me. Well, he figures it out only because he gets it from the... Let's say, he believes with some certainty that he's going to be called the same as the machine. So he gets it from the hostname file. I know what I showed you a little while ago. But you're going to see that that won't be enough because we're also going to create it so that it's called stork. So he's going to know himself by two names. So we say yes. He finishes installing.

The ideal thing here would have been... But again, we'd have to modify the entire lab platform. The ideal thing would have been to make it universally accessible, so the host name wouldn't be cli.grp1, but instead, for example, could be called stork. That would have been a riskier step. Configure everything with that universalized hostname. It can be done. The only thing is that it will take more time.

So, here we are going to touch on the configuration of postfix itself. See, I don't use nano, I use vi. Here we are representatives of two philosophical schools separated by a philosophical chasm, I would say. They're like the Capulets.

Nicolás: I think they are quite contemporaneous, but there is a philosophical question there.

Carlos: Well, we're going to have to change two or three variables here. One is called myDestination. In fact, that's an age-old discussion we have in some technical groups. Cristian is in one, and all that. I'm in the technical one. Then there are those

who are just walking. Of course. Once the good snow starts rolling down the mountain, this is going to turn into a kind of fundamentalism.

Nicholas: Tell them about the day I had to apologize while you were making the demo.

Carlos: Yes, yes. Yes, a lot of people got furious... During the pandemic, we had a talk about this. No, it wasn't about DNSSEC, but it was the same lab, the same infrastructure. And it was all virtual. And it was for Brazil. So, like everything in Brazil, there were 600 people connected to the stream. And when he opened the nano in the chat...

Nicholas: The face uses nano! The face uses nano!

Carlos: It was in Portuguese, but they called me everything except cousin. Impressively impressive.

Nicolás: Do you know how these arguments get out? I invite you all to do a simple Google search to find out which one is the most used. And that's where the arguments end.

Carlos: It's perfect for kids. The thing is, nano is useless.

Nicholas: Of course. No, there aren't any worse ones I've seen. There are worse ones I've seen.

Carlos: Yeah, sure. I never got around to Emacs, for example. And not for lack of trying.

My domain... I changed it here, I changed it there. Note that what I'm adding is precisely all the strings by which the machine must recognize itself. We also include this later in the documentation. Yes. It's specific to Postfix. Each client and each email server is configured differently, so to speak. But in the case of Postfix, you have to modify two or three things, which basically involve adding the domain name to that universalized one we want to use to identify our email server.

In our case we are going to add a domain name that is... Note that the one we configured by default and that will also work is... cli.grp1.uamvdt.labstraining And now we also add that the mail server is also called cigüeña.grp1.uamvdt.labstraining

Nicholas: One day when we want to beat ourselves up, we have to do this with SendMail. Also.

Carlos: And basically, those are the configuration changes you need to make. There you have it. This should be enough. I added one here, though no one asked me why. This one says My Networks. What's going on? The default configuration for all these things in general in today's world, where you connect to an internet server and you're immediately attacked. By default, you install servers for things, and they only listen on the localhost interfaces. Those called 127 something something. They don't receive internet connections, and that's for security reasons. It's designed this way so that when you open it to the internet, it's a conscious action. And not like you installed something, went out to make a cup of coffee, and in the middle of it, you have three Russians living inside the machine.

Nicholas: There were things that worked, but that was just the way it was at the time.

Carlos: So right now, what I'm telling him is that any interface on 100.100/16 is an interface he can listen to traffic on. I'm taking that conscious step of opening him up to this mock-up of the internet in some way.

Nicolás: I'm talking about this for those who aren't familiar with security issues. Anything you do on the internet, especially those things that are highly visible or highly exposed, like DNS servers, email servers, etc. There are literally millions of attack services looking for machines they can use to carry out other attacks. Not to harm you. Normally, the owners of those machines or devices never realize they have them installed and that they're part of an attack. Because they're used to attack a third party. But between the time you set up an open DNS server, for example, or an email server open to the world without a reasonably secure configuration and being attacked, it's likely a matter of minutes. Minutes.

Carlos: What's really in vogue now, more than attacking third parties—even that's lost its appeal—is mining crypto. All the latest machines that you've seen compromised in some way in the last four or five years, which hasn't been many, but they've all been used for crypto mining. It's impressive.

Nicholas: Now I have to explain what crypto mining is, you see?

Carlos: Okay, I think we're there. So now what I'm going to do is open... Did you install the client?

Nicholas: I installed the client.

Carlos: So what Carlos did now was install the email client. Actually, both email clients, because we installed two. We installed Mutt and Pine. Mutt and Pine are two email clients that are used via the command line. Not like Outlook, which has a beautiful graphical environment, but here, like we're in a terminal.

Nicolás: Yes, in a more sophisticated version on this side we could install webmail or something like that, but that would be a different matter.

Carlos: I have an idea to give you with that.

Nicolás: Okay. We installed the MTA, the email agent. There it is. They're all on the same machine. So now Carlos is going to use one of the email clients. If I'm not mistaken, Mutt.

Carlos: I'm going to start with the pine. Because that's the one I tried a little while ago.

Nicolás: It's going to start with the pine. Send an email to the user to the email address nicolas@. You're going to use stork. Stork. nicolas@stork dot grp1 dot uamvd dot tlabs dot training. Which has the name of the universalized email address. Nicolás contains the accent. And it has the universalized domain for the lowest-level part of the domain, which is stork. So it's going to send an email there and it's going to use the pine to check whether or not it receives that email. And then it's going to use the mutt client to send another email to that same address and see.

Carlos: Nicolás will be at the bottom. Martínez at the top. And there I'll open the pine. And I'll have to make a couple of configurations in the pine. And those are the two users.

Nicolás: Is Raúl around? Or did Raúl go out? Oh. Of course. He's fine. Because he had asked about the pine tree this morning.

Carlos: Okay. All I have to do is come here and tell him where the inbox is. The inbox is in var mail. Which one is this? This is Martínez. They're two universalized users. I made a mistake again. Martínez with an accent there. He's sending to Nicolás with an accent there. Mail. Martínez. See, he's showing me it's kind of filthy there, but it works. Maybe it's not that filthy.

Nicholas: Do you know what UTF-8 might be? I'd have to look into that.

Carlos: Okay. So, for this one, I say exit. Commit changes. And we go to the Message Index. And I don't have email. No one has written to me yet. I'm alone there. So, we have to do the same thing with this one. We've created mailboxes with Chinese names before, but that's it.

Nicolás: So, it took me a while to find... Nico has that text file with some examples in Chinese that I don't... It's going to be impossible to write in Chinese, so... It's more picturesque, but... It's definitely more picturesque. It's harder to write. It's easier to fool the audience, right? Because you put two boxes that are the same. They're not exactly the same, but they differ by a little line. Like, one character. So that just someone knows.

Carlos: Sure. Yeah, well, here it goes. Smartmail. Nicholas.

Nicolás: Joking aside, you don't know how difficult and problematic it is to conduct these training sessions in English for people who speak Chinese or Mandarin. That can be tricky. Not for us to speak it, let's say. I could teach them in Yoruba, but I don't know if that's useful. It takes us twice as long to do everything.

Carlos: Well, that's where we're at. So, let's send an email to Martínez. Compose Martínez at... Ah, now the same thing happens to me, let's see if... stork, there it is. GRP1, what was it? UAMVD. UAMVD. Okay, we're... Training... And... Let's put something here... What Puny Code is it? Hello? Hello should have a... He should send some passwords and some umlauts there. And then we tell him to send it. And if all this worked at some point... There. Nicolás sent it to Martínez. I mean, Nicolás sent it to Martínez...

Nicholas: Yes.

Carlos: I'm just checking if I'm confused. Yes. Oh, wait, it bounced back. Let's see what I did wrong. What the hell is he saying he couldn't send the email? What do I know? Ah, loops back to my cell.

Nicholas: Do you remember? This is a classic of these things. What was this like? No, it's not something else. It's the... What is it like? No, no, it's not something else.

Carlos: It's a Postfix configuration issue. Maybe I'm missing something. I'm stuck here with something strange. Let's see. This one appears twice. What happened

here? CLI grp1 localhost grp1. The stork. There it is. And let's see if I'm missing something else. Hahaha. What do I have here? Yes, yes, yes, yes. Where is it looking at me? Does it forget names well from there?

Nicholas: Oh, that's it. He doesn't forget names well? Did you notice? Look. That comes up if I put... We're doing it without MX. We're not doing it without MX because...

Carlos: There it goes. Yes. Yes. What was I going to tell you? Let's see. 100 100 1 2. That's fine, right? And the postfix? The postfix? I don't have to enter the address. Oh, I put 100 100 0 0 slash 16, which should be more than enough. Okay, I fixed a little thing there. grp1 grp1 grp1. Let's see, let's try it now. grp1 uamvd tlabs training. Oh? See? There it is. Delivered. I think so. I think so. That line is already wrong. How come they bring me a strange character? There it is, well, there it came from mutt.

So now, for example, we can forward this to Martínez. Oh, what a fighter! Could it be? For me, it's an issue when... It's the client, let's say the client, who's mixing it up with the domain name and thinks it's the loop, so he doesn't even send it. Let's try with mutt. It doesn't even get passed to the post, look. Let's do echo, testing from Nicolás.

Nicholas: Oh, I have so much love for the pine, it failed me.

Carlos: I just sent the email using mutt by typing it in the command line instead of using pine to send it. Bingo! I bet it arrived. There it is. It works for receiving, right?

Nicolás: Well, that's exactly how difficult it is for each... Yeah, exactly... universal acceptance. It's true that I went vintage, too, right? Using Pine was kind of a lot. And there's something that Carlos and I talked about today, as far as I remember, and that's why, because we're stubborn, we must be the only ones who do tests when we're doing a lab presentation, but... Pine didn't have 100% universal acceptance support until a while ago. That's why we used Mutt, so to speak. Maybe it has something to do with that. Maybe it does, because it's working perfectly for reception.

Carlos: It works for receiving. And in fact, it shows me—look, the FROM shows it correctly. I'd have to see if that's what it is, so to speak. Maybe it's been updated in the last 22 years.

Nicholas: I doubt the pine has been updated but it could very well be.

Carlos: So, that would be the demonstration. Regarding the email interface we're using, we're using something very basic that can be configured in the terminal. For a next iteration of this, we could install webmail or something like that, which is fairly easy. Note that it has its complications and must be done very carefully. You have to use the tools, these web interfaces like punicoder.com to create the strings. And you have to test a lot. But the good thing is that it works.

And these things are the hardest part, once you get them working, the first one is the hardest. Afterward, they become more or less easy. And also, the idea, as we said at the beginning, is to show that it's not that complicated to do. We're doing it here in a presentation; we haven't tested it 200,000 times either. We've done it a few times, and in production, it's not very different from this, so to speak.

The DNS scheme we built is, I think, a bit too realistic for a demonstration. For email, we could have separated the MTA on one machine and the client on another. Perhaps if we had done that, we wouldn't have this loopback problem. And you can try it out at home. And well, the idea is to show... Postfix is one of the most widely used email clients, too.

So, everyone who has Postfix and doesn't have universal acceptance configured could configure it to work. It's just one or two lines of configuration. Because most of the configuration is already there for that. We simply added the configuration because we wanted to name the server's domain name "Stork." But if your email server doesn't have a universalized domain, you practically don't have to do anything. The same goes for the email client. Well, now I'm doing my homework: verifying if Pine has partial support, which is why it can't send but can receive, or if it actually has 100% support, but it's a configuration issue for us since we installed everything on the same machine, and Pine doesn't play well with that, let's say.

Nicolás: Yes, this one, but MUTT works perfectly for sending, and MUTT says it has 100% universal acceptance support. If you use Outlook, for example, Outlook is also one of those that currently has 100% universal acceptance support. So, if you have an ecosystem where your email server and MTA are Postfix, your clients use Outlook, for example, at the corporate level, but not at the corporate level, and you configure this and the DNS appropriately, you'll have an environment that can perfectly support universal acceptance.

Remember that if you have any system or user interface to create a mailbox etc. that uses a web form for example or a user in the password whatever is good, you are also going to have to make that web form so that when I put Nicholas with an accent because I want to create my mailbox, it lets me create it, it doesn't tell me no, the accent is not a valid character for a mailbox name because if not, everything you did in postfix and everything we did until now doesn't make sense, let's say because the user interface doesn't let you create it.

But well, it was a little to show that later we leave all this, we are going to clean it up without all the errors and without all the tests and we leave it for those who are more technical to want to try this if they know how to use this virtual box, for example, 3, 4, 5 virtual machines are created on their machine, they install all these things and following the letter of the laboratory they can reproduce 100% exactly what we did here without any problem.

Well, they're going to have a problem: they're going to have to be able to use a domain to create the DNS domain, so to speak. But that's easy and not very expensive to do.

Carlos: Yes, there's even a way to do it without that, which is to create all the names in the etc. hosts of the machines you use. It's kind of a hassle, but if you don't have a domain...

Nicolás: Nothing could be further from the truth. It's wrong for me to say that because I don't sell domains. They can do it that way too.

Carlos: And if they can't use any other virtualization environment, Docker LXD or whatever...

Nicholas: If not, they can always use .corp and contribute to AS102 traffic. That's fine too.

Carlos: Well, what I did last time was that mutt not only can be used completely as a shell, but it has an interface that looks a little more modern. Instead of 25 years old, I'd say it's only 17. It's similar to Pine, and I sent an email and it arrived. So, it's clearly an issue that Pine doesn't support. It confirms that Pine doesn't have 100% support.

Nicolás: Raúl, we tried to do this for you, trying with the PIN, but it didn't work out. The bottom line is that we're sure it can send and we're not sure if it's us or the PIN, so to speak.

Carlos: You have to try it. You have to try it.

Nicholas: Okay, very good. Well, that's it. I don't know if you have any questions or comments. Also.

Carlos: There's already a question. Wait, because there are people far away, so they can listen too.

Participant: There you go. How are you? Good afternoon. Do you know if the Postfix alias file accepts international names?

Carlos: That's a tremendous question. We'd have to try it, but I'd say 99% yes. Yes. I told you in the compliance list, how do you translate compliance?

Nicholas: What? Compliance. Compliance is compliance. It's like a checklist of how compliant they are.

Carlos: They're clearly compliant. The problems, right? We've done some damage in English, but we still don't know how to translate some words. Well, on the list of compliance levels for universally accepted software, as I've told you, there's a very easy classification: level 1, level 2. Level 1 means I can do some things but not others. Level 2 means I can do everything. On the list, Postfix is classified as level 2 support. That means anything with a domain name should be able to be defined as universally accepted. So, it's almost certain that it will. I can't say for 100% sure because I've never tried it, but if not, it wouldn't be level 2. That's a good question.

Nicholas: It's a good one for... It's a good one that we should do a little test of so we can show it. If you want to reproduce this atmosphere...

Carlos: If you try it, let us know how it went. We'll be in touch because it's a good thing to try. We'd have to test whether it works well on both sides, because the Postfix or SendMail alias tables are also key-value tables. Of course. In which you have a name that maps to one or more names. We'd have to see if the internationalized characters work on both sides.

Participant: There you go. Yes, yes. Perfect, thank you very much.

Nicolás: At the DNS level, for example, yes, at the DNS level. Any record with any Puny Code. A record like... That's the equivalent of an alias for the DNS. An MX record. A record... Even a DNS record. In other words, you can have a name server that's universally accepted. In fact, of the 1,200 generic top-level domain names that Rodrigo was talking about today, there are quite a few that are universalized and have a Puny Code as part of the encoding, so to speak.

Carlos: Well, I'll stop sharing so I can release this.

Nicholas: Well, thank you all very much.